Amendments to the Claims:

This listing of claims will replace all prior versions, and listings of claims in the application:

Listing of Claims:

1 - 21. (Canceled)

22. (New): A method for accessing a first datum stored in a memory, the first datum comprising one or more first bytes and one or more second bytes, the method comprising:

performing a first memory access to read out the first bytes;

performing a first rotation operation on the first bytes to produce first rotated datum;

replacing portions of the first rotated datum with binary zeroes by performing a masking operation on the first rotated datum;

storing the first rotated datum to a first data store;

performing a second memory access to read out the second bytes;

performing a second rotation operation on the second bytes to produce second rotated datum;

replacing portions of the second rotated datum with binary zeroes by performing a sign extension operation on the second rotated datum;

storing the second rotated datum to a second data store;

performing an OR operation of the content of the first data store and the second data store to produce the first datum; and

storing the first datum to a third data store,

wherein the first datum as stored in the memory can be an aligned datum or an unaligned datum.

23. (New): The method of claim 22 wherein the first, second, and third data stores are 64-bits in length.

24. (New): The method of claim 23 wherein the datum is one of a 16-bit datum, a 32-bit datum, and a 64-bit datum.

25. (New): The method of claim 22 wherein the first and second bytes are stored in consecutive memory locations in the memory.

26. (New): The method of claim 22 further comprising performing a sign extension operation on the first rotated datum subsequent to performing the masking operation, to extend a sign bit of the first rotated datum.

27. (New): A method for accessing a datum stored in a memory, wherein the datum as stored in the memory can be an aligned datum or an unaligned datum, the method comprising:

reading one or more first bytes from the memory, the first bytes being a first portion of the datum;

performing a first rotation operation on the first bytes to produce first rotated bytes;

replacing portions of the first rotated bytes with binary zeroes by performing a masking operation to produce first aligned bytes;

performing a sign extension operation on the first aligned bytes to extend a sign bit of the datum represented by the first aligned bytes to produce extended aligned bytes;

storing the extended aligned bytes to a first data store;

reading one or more second bytes from the memory, the second bytes being a second portion of the datum;

performing a second rotation operation on the second bytes to produce second rotated bytes;

replacing portions of the second rotated bytes with binary zeroes by performing a sign extension operation to produce second aligned bytes;

storing the second aligned bytes to a second data store; and

performing an OR operation of the first aligned bytes and the second aligned bytes to produce the first datum.

28. (New): The method of claim 27 wherein the first and second bytes are stored in consecutive locations in the memory.

29. (New): The method of claim 27 further comprising storing the first datum to a third data store subsequent to performing the OR operation.

30. (New): The method of claim 29 wherein the first, second, and third data stores are 64-bits in length.

31. (New): The method of claim 30 wherein the datum is one of a 16-bit datum, a 32-bit datum, and a 64-bit datum.

32. (New): A method for accessing a datum stored in a memory comprising:

accessing the memory to read out a first portion of the datum, the first portion comprising one or more first bytes;

performing a first rotation operation on the first bytes to produce rotated first bytes, wherein a rotate distance of the first rotation operation is based on whether the datum stored in the memory is stored in Big Endian order or in Little Endian order;

replacing portions of the rotated first bytes with binary zeroes to produce aligned first bytes;

storing the aligned first bytes to a first data store;

accessing the memory to read out a second portion of the datum, the second portion comprising one or more second bytes;

performing a second rotation operation on the second bytes to produce rotated second bytes, wherein a rotate distance of the second rotation operation is based on whether the datum stored in the memory is stored in Big Endian order or in Little Endian order;

replacing portions of the rotated second bytes with binary zeroes to produce aligned second bytes;

storing the aligned second bytes to a second data store; and

performing an OR operation of the content of the first data store and the second data store to produce the datum.

33. (New): The method of claim 32 wherein the datum as stored in the memory can be an aligned datum or an unaligned datum.

34. (New): The method of claim 32 wherein the step of replacing portions of the rotated first bytes with binary zeroes includes performing a masking operation.

35. (New): The method of claim 34 wherein the step of replacing portions of the rotated second bytes with binary zeroes includes performing a sign extension operation.

36. (New): The method of claim 32 wherein the datum is one of a 16-bit datum, a 32-bit datum, and a 64-bit datum.

37. (New): The method of claim 32 further comprising storing the datum to a third data store subsequent to performing the OR operation.

38. (New): A data processor for accessing a stored datum stored in a memory, the stored datum comprising one or more first bytes and one or more second bytes, the circuit comprising:

a data input for receiving a portion of the first datum from the memory;

a rotator circuit coupled to the data input and having a rotator output, the rotator circuit which produce a first rotated datum and a second rotated datum,

a zero-filling circuit coupled to the rotator output and configured to replace one or more bit positions of a datum on the rotator output with binary zeroes, the zero-fill circuit which performs a sign extension operation to fill portions of the first rotated datum with binary zeroes thereby producing a first aligned datum; and performs a sign extension operation to fill portions of the second rotated datum with binary zeroes thereby producing a second aligned datum; and

a logical operation circuit which performs OR operation of the content of the first aligned datum and the second aligned datum, the OR result being the first datum,

wherein the first datum as stored in the memory can be an aligned datum or an unaligned datum.

C1

39. (New): The processor of claim 38 wherein the first, second, and third data stores are 64-bits in length.

40. (New): The processor of claim 38 wherein the zero-fill circuit further performs a sign extension operation on the first rotated datum subsequent to performing the masking operation, to extend a sign bit of the first rotated datum.

41. (New): The processor of claim 38 wherein the OR result is stored to a third data store.

42. (New): A CPU having a circuit configured to access a datum stored in a memory comprising:

a data input for receiving a portion of the datum from the memory;

a rotator circuit coupled to the data input and having a rotator output, the rotator circuit configured to perform a rotation operation of data on the input and to produce rotated data at the output;

a distance control circuit having a control signal coupled to the rotator circuit, wherein the control signal is dependent on whether the datum that is stored in the memory is stored in Big Endian order or in Little Endian order, wherein an amount of rotation performed by the rotator circuit is dependent on the control signal; and

a zero-filling circuit coupled to the rotator output and configured to replace one or more bit positions of a datum on the rotator output with binary zeroes.

43. (New): The circuit of claim 42 wherein the datum stored in the memory can be an aligned datum or an unaligned datum.

44. (New): The circuit of claim 42 wherein the datum stored in the memory comprises a first portion of one or more bytes of data and a second portion of one or more bytes of data, wherein the first portion is processed by the circuit to produce a first result and the second portion is processed by the circuit to produce a second result, wherein an OR operation performed on the first result and the second result produces the datum stored in the memory.

45. (New): A data processor system comprising:

a memory, wherein a first datum is stored in the memory, wherein the first datum can be an aligned datum or an unaligned datum;

a data input for receiving a portion of the first datum from the memory;

a rotator circuit coupled to the data input and having a rotator output, the rotator circuit operable to produce a first rotated datum and a second rotated datum,

a zero-filling circuit coupled to the rotator output and configured to replace one or more bit positions of a datum on the rotator output with binary zeroes, the zero-fill circuit operable to perform a sign extension operation to fill portions of the first rotated datum with binary zeroes thereby producing a first aligned datum, the zero-fill circuit operable to perform a sign extension operation to fill portions of the second rotated datum with binary zeroes thereby producing a second aligned datum; and

a logical operation circuit which performs an OR operation of the first aligned datum and the second aligned datum, wherein a result of the OR operation is the first datum.

46. (New): A data processor system comprising:

a memory, wherein a first datum is stored in the memory, wherein the first datum can be an aligned datum or an unaligned datum;

a data input for receiving a portion of the first datum from the memory;

a rotator circuit coupled to the data input and having a rotator output, the rotator circuit configured to perform a rotation operation of data on the input and to produce rotated data at the output;

a distance control circuit having a control signal coupled to the rotator circuit, wherein the control signal is dependent on whether the first datum is stored in the memory in Big Endian order or in Little Endian order, wherein an amount of rotation performed by the rotator circuit is dependent on the control signal; and

a zero-filling circuit coupled to the rotator output and configured to replace one or more bit positions of a datum on the rotator output with binary zeroes.

47. (New): The data processor system of claim 46 wherein the datum stored in the memory comprises a first portion of one or more bytes of data and a second portion of one or more bytes of data, wherein the first portion is processed by the circuit to produce a first result and the second portion is processed by the circuit to produce a second result, wherein an OR operation performed on the first result and the second result produces the datum stored in the memory.